In re Application of PARKES et al.
Application No. 09/779,763

Amendments to the Claims

1. (Currently Amended) A method of performing a task on a computer system having a cache, wherein the task comprises a plurality of sub-tasks, the method comprising:

(a) instantiating a plurality of stages, the plurality of stages comprising at least one stage for each of the plurality of sub-tasks, each stage comprising:

a holding area; and

a scheduling policy;

~~(a)~~ (b) receiving a plurality of requests for the task;

~~(b)~~ (c) for each of the requests, storing in a work packet data for performing a sub-task of the task;

~~(c)~~ (d) storing each work packet in a the holding area of the at least one stage for the sub-task;

~~(d)~~ (e) executing the sub-task on each of the work packets in the holding area in accordance with the scheduling policy of the at least one stage for the sub-task while refraining from executing other sub-tasks of the task ~~procedure~~, thereby maintaining locality of instructions and data in the cache; and

~~(e)~~ (f) repeating steps c, d and e ~~b, c and d~~ for each sub-task of the task until all of the sub-tasks of the task are completed for each request.

2. (Original) The method of claim 1, wherein each sub-task has a type of work packet defined for it, wherein the work packet defined for the sub-task includes data and functions for performing the sub-task.

3. (Original) The method of claim 1, wherein the holding area is a queue.

4. (Original) The method of claim 1, wherein the holding area is a stack.

5. (Original) The method of claim 1, wherein at least one of the executed work packets is a parent work packet, the method further comprising: creating a child

work packet for the parent work packet; and performing a sub-task of the plurality of sub-tasks on the child work packet.

6. (Canceled)

7. (Currently Amended) The method of claim 1, wherein step (c) ~~(b)~~ further comprises storing in the work packet a pointer to the data for performing the sub-task.

8. (Original) The method of claim 5, wherein the sub-task performed on the child work packet is different from the sub-task performed on the parent work packet.

9. (Original) The method of claim 5, wherein the sub-task performed on the child work packet is the same as the sub-task performed on the parent work packet.

10. (Original) The method of claim 5, wherein the sub-task being performed on the parent work packet is halted until the sub-task being performed on the child work packet is completed.

11. (Original) The method of claim 5, wherein the sub-task being performed on the parent work packet is halted until a predefined event occurs.

12. (Currently Amended) A method of performing a task on a computer system having a cache, wherein the task comprises a plurality of sub-tasks, the method comprising:

creating an instance of a stage for each sub-task, wherein the stage comprises: ~~has an associated~~

a holding area; and,

a scheduling policy;

placing one or more work packets in the holding area, each work packet corresponding to an iteration of the sub-task required for the task and ~~, each work packet~~ containing data for performing the sub-task; and,

performing the sub-task on each work packet in the holding area of the stage in accordance with the scheduling policy of the stage ~~so that the sub-task is repeatedly performed~~, thereby maintaining data locality in the cache for the sub-task.

13. (Original) The method of claim 12, wherein the stage permits an instance of itself to be created on only a single processor in the computer system at a time.

14. (Currently Amended) The method of claim 12, wherein the stage includes a local data area, and wherein the stage regulates which processor is permitted to create an instance of it based on the part of the local data area ~~is~~ required to be accessed.

15. (Original) The method of claim 12, wherein, for at least one of the stage instances created, there is at least one work packet that is a parent work packet, the method further comprising: creating a child work packet for the parent work packet; sending the child work packet to another stage instance; and; at the other stage instances, performing a sub-task of the plurality of sub-tasks on the child work packet.

16. (Original) The method of claim 12, wherein the holding area includes a stack and a queue, and the method further comprises: for each work packet received by at least one stage instance, putting the work packet in the stack if the work packet originated from the processor on which the instance of the stage is created, and putting the work packet in the queue if the work packet originated from another processor.

17. (Original) The method of claim 12, wherein each work packet contains at least one pointer to data for performing the sub-task.

18. (Canceled)

19. (Currently Amended) A system for executing a procedure on a computer, wherein the procedure is divided into a plurality of sub-tasks, the system comprising:

a computer-readable medium having stored thereon ~~there on~~ a plurality of work packets, each work packet including data usable to perform an iteration of a sub-task of the plurality of sub-tasks;

a computer-readable medium having stored thereon a plurality of stages, there being at least one stage for each sub-task, each stage comprising:

a holding area for holding a batch of the plurality of work packets; and,

a scheduling policy; and,

a processor for identifying a stage of the plurality of stages and performing an iteration of the stage's sub-task on each of the batch of work packets in accordance with the scheduling policy of the stage, thereby maintaining ~~the~~ a locality of data in a cache of the processor.

20. (Original) The system of claim 19, wherein the processor is one of a plurality of processors and wherein at least one stage of the plurality is instantiated on at least two of the plurality of processors.

21. (Original) The system of claim 19, wherein the holding area of the identified stage includes a queue and a stack, wherein the queue is for holding work packets that originated from processors other than the one on which an instance of the identified stage is created, wherein the stack is for holding work packets that originated from the processor on which the instance of the stage is created.

22. (Original) The system of claim 19, wherein the identified stage has a local data area that is divided into sections, and wherein the processor determines whether to perform the sub-task of the stage based on the section of the local data area to which each work packet of the batch will require access.

23. (Original) The system of claim 19, wherein each work packet contains instructions necessary to perform the sub-task of the stage at which it is held.

24. (Original) The system of claim 19, wherein each work packet contains at least one pointer to instructions necessary to perform the sub-task of the stage at which it is held.

25. (Original) The system of claim 19, wherein each stage contains instructions necessary to perform its sub-task.

26. (Original) The system of claim 19, wherein the processor is one of a plurality of processors, wherein the holding area of each stage is one of a plurality of holding areas for the stage, and wherein each of the plurality of holding areas of a stage is associated with one of the plurality of processors.

27. (Original) The system of claim 19, wherein the holding area is a queue.

28. (Original) The system of claim 19, wherein the holding area is a priority queue.

29. (Original) The system of claim 19, wherein the holding area is a stack.

30. (New) A computer-readable medium having stored thereon computer-executable instructions for a method of performing a task on a computer system having a cache, wherein the task comprises a plurality of sub-tasks, the method comprising:

(a) instantiating a plurality of stages, the plurality of stages comprising at least one stage for each of the plurality of sub-tasks, each stage comprising:

a holding area; and

a scheduling policy;

(b) receiving a plurality of requests for the task;

(c) for each of the requests, storing in a work packet data for performing a sub-task of the task;

(d) storing each work packet in the holding area of the at least one stage for the sub-task;

(e) executing the sub-task on each of the work packets in the holding area in accordance with the scheduling policy of the at least one stage for the sub-task while refraining from executing other sub-tasks of the task, thereby maintaining locality of instructions and data in the cache; and

(f) repeating steps c, d and e for each sub-task of the task until all of the sub-tasks of the task are completed for each request.

31. (New) The computer-readable medium of claim 30, wherein each sub-task has a type of work packet defined for it, wherein the work packet defined for the sub-task includes data and functions for performing the sub-task.

32. (New) The computer-readable medium of claim 30, wherein the holding area is a qucue.

33. (New) The computer-readable medium of claim 30, wherein the holding area is a stack.

34. (New) The computer-readable medium of claim 30, wherein at least one of the executed work packets is a parent work packet, the method further comprising: creating a child work packet for the parent work packet; and performing a sub-task of the plurality of sub-tasks on the child work packet.

35. (New) The computer-readable medium of claim 30, wherein step (c) further comprises storing in the work packet a pointer to the data for performing the sub-task.

36. (New) A computer-readable medium having stored thereon computer-executable instructions for a method of performing a task on a computer system having a cache, wherein the task comprises a plurality of sub-tasks, the method comprising:

creating an instance of a stage for each sub-task, wherein the stage comprises:

a holding area; and,

a scheduling policy;

placing one or more work packets in the holding area, each work packet corresponding to an iteration of the sub-task required for the task and containing data for performing the sub-task; and,

performing the sub-task on each work packet in the holding area of the stage in accordance with the scheduling policy of the stage, thereby maintaining data locality in the cache for the sub-task.

37. (New) The computer-readable medium of claim 36, wherein the stage permits an instance of itself to be created on only a single processor in the computer system at a time.

38. (New) The computer-readable medium of claim 36, wherein the stage includes a local data area, and wherein the stage regulates which processor is permitted to create an instance of it based on the part of the local data area required to be accessed.

39. (New) The computer-readable medium of claim 36, wherein, for at least one of the stage instances created, there is at least one work packet that is a parent work packet, the method further comprising:

creating a child work packet for the parent work packet;

sending the child work packet to another stage instance; and;

at the other stage instances, performing a sub-task of the plurality of sub-tasks on the child work packet.

40. (New) The computer-readable medium of claim 36, wherein the holding area includes a stack and a queue, and the method further comprises: for each work packet received by at least one stage instance, putting the work packet in the stack if the work packet originated from the processor on which the instance of the stage is created, and putting the work packet in the queue if the work packet originated from another processor.

41. (New) The computer-readable medium of claim 36, wherein each work packet contains at least one pointer to data for performing the sub-task.